Скрипты всех решений прикреплены

Создадим скриптом таблицы R и T:

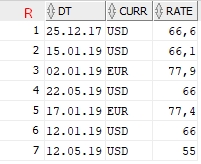
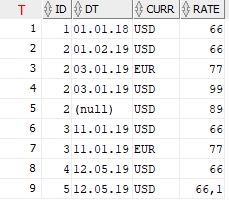
Таблица R  


Таблица T  


**Задача 1**

Напишите запрос, который вернет все записи из таблицы t (колонки id, dt, curr, rate), в которых идентификатор id дублится.

*Представьте не менее трех вариантов решения.*

Решение 1 – через JOIN c условием:

SELECT t.id, t.dt, t.curr, t.rate FROM t INNER JOIN  
(SELECT t.id, COUNT(t.id) AS id\_count FROM t  
GROUP BY t.id) t\_count  
ON t.id = t\_count.id  
WHERE t\_count.id\_count > 1;

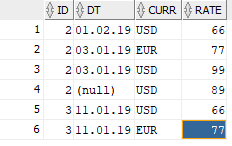
Решение 2 – через JOIN с подзапросом с HAVING:

SELECT t.id, t.dt, t.curr, t.rate FROM t INNER JOIN  
(SELECT t.id, COUNT(t.id) AS id\_count FROM t  
GROUP BY t.id  
HAVING COUNT(t.id) > 1) t\_count  
ON t.id = t\_count.id;

Решение 3 – через условие IN и подзапрос в HAVING:

SELECT t.id, t.dt, t.curr, t.rate FROM t  
WHERE t.id IN  
(SELECT t.id FROM t  
GROUP BY t.id  
HAVING COUNT(t.id) > 1)

Во всех 3-х решениях получается следующий результат:



**Задача 2**

Напишите запрос, который вернет только уникальные по id строки (колонки id, dt, curr, rate). Оставить необходимо ту строку в рамках id, у которой dt максимальная. В случае строк с идентичными максимальными датами можно оставить любую из них.

*Представьте не менее трех вариантов решения.*

Начало решения:

Вообще, довольно легко удалось получить записи, удовлетворяющие условиям:

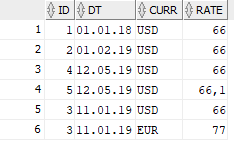
SELECT t.id, t.dt, t.curr, t.rate FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt

Получаем следующий результат:



**У нас есть дубли.** Как же нам осуществить выбор «любого из них»? Вот тут начинается самое интересное.

Решение 1 – через ввод нумерации, условие **max** для этого номера:

SELECT not\_unic.id, not\_unic.dt, not\_unic.curr, not\_unic.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

INNER JOIN

(SELECT **max**(not\_unic.num) as num, not\_unic.id FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

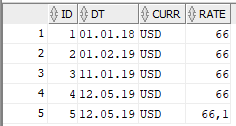
(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

GROUP BY not\_unic.id) **t\_max**

ON **t\_max**.num = not\_unic.num



Решение 2 – через ввод нумерации, условие **min** для этого номера (можно же любое решение):

SELECT not\_unic.id, not\_unic.dt, not\_unic.curr, not\_unic.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

INNER JOIN

(SELECT **min**(not\_unic.num) as num, not\_unic.id FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

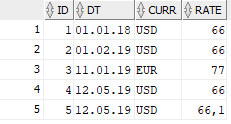
(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

GROUP BY not\_unic.id) **t\_min**

ON **t\_min**.num = not\_unic.num



Решение 3 – а ведь нумерацию можно ввести к первоначальнмоу варианту, а не к промежуточному, и сразу применить условие **max** номера (понял в процессе, что можно покороче сделать), а потом обойтись без джойнов, а использовать простой IN:

SELECT t\_num.id, t\_num.dt, t\_num.curr, t\_num.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t) t\_num

WHERE t\_num.num IN

(SELECT **max**(not\_unic.num) as num FROM

(SELECT t\_num.num, t\_num.id, t\_num.dt, t\_num.curr, t\_num.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num,

t.id, t.dt, t.curr, t.rate

FROM t) t\_num

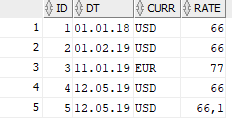
INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t\_num.id = t\_max\_date.id AND t\_num.dt = t\_max\_date.dt) not\_unic

GROUP BY not\_unic.id)



Решение 4 – тоже, что и в 3, но с **min**. Можно множить варианты скрипта решения, меняя функции, до безумия плодотворно.

**Задача 3**

Напишите запрос, который удалит лишние строки в тех случаях, когда идентификатор id дублится. Оставить необходимо ту строку в рамках id, у которой dt максимальная. В случае строк с идентичными максимальными датами можно оставить любую из них.

*Представьте не менее двух вариантов решения.*

Решение:

Взять задачу 2 и перезаписать таблицу. Ключа то в таблице нет – я вижу только такой способ. Или через функции, но это сложно, я не успею, пока буду уразумевать функции. Приступим к решению.

Решение 1:

Рассмотрим скрипт с решением 3 задачи 2 (он покороче и попонятнее):

CREATE TABLE ark AS (

SELECT t\_num.id, t\_num.dt, t\_num.curr, t\_num.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t) t\_num

WHERE t\_num.num IN

(SELECT max(not\_unic.num) as num FROM

(SELECT t\_num.num, t\_num.id, t\_num.dt, t\_num.curr, t\_num.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num,

t.id, t.dt, t.curr, t.rate

FROM t) t\_num

INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t\_num.id = t\_max\_date.id AND t\_num.dt = t\_max\_date.dt) not\_unic

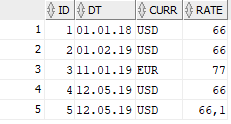
GROUP BY not\_unic.id)

);

DROP TABLE t;

RENAME ark TO t;

SELECT \* FROM t;



Решения 2, 3, 4 … n – вариации с различными c решением задачи 2. Укажу парочку, раз уж порядок изложения такой для «любого» по принципу min и max:

CREATE TABLE ark AS (

SELECT not\_unic.id, not\_unic.dt, not\_unic.curr, not\_unic.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

INNER JOIN

(SELECT min(not\_unic.num) as num, not\_unic.id FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

GROUP BY not\_unic.id) t\_min

ON t\_min.num = not\_unic.num

);

DROP TABLE t;

RENAME ark TO t;

SELECT \* FROM t;

CREATE TABLE ark AS (

SELECT not\_unic.id, not\_unic.dt, not\_unic.curr, not\_unic.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

INNER JOIN

(SELECT max(not\_unic.num) as num, not\_unic.id FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.id, max(t.dt) as dt FROM t

GROUP BY t.id) t\_max\_date

ON t.id = t\_max\_date.id AND t.dt = t\_max\_date.dt) not\_unic

GROUP BY not\_unic.id) t\_max

ON t\_max.num = not\_unic.num

);

DROP TABLE t;

RENAME ark TO t;

SELECT \* FROM t;

А ведь можно было не пересоздавать таблицу, а просто чистить и перезаписывать данные (я так делаю в задании 4 и 5). Такое решение ко мне пришло после задачи 4, не буду его приводить, т.к. это вполне подпадает под концепцию «множить решения, меняя команды». Суть SEELCT и перезаписи остаётся прежней. Вот если бы я владел в oracle циклами, как в python, вот это был бы принципиально новый способ.

**Задача 4**

Необходимо обновить rate в таблице t данными rate таблицы r для t.id = 5. Из таблицы r берем информацию только за даты 17.01.2019 и 12.05.2019. Связь по ключу curr.

- Обновить существующие записи.

- Добавить новые записи (id = 5).

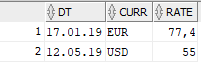
*Представьте не менее двух вариантов решения.*

Насколько я понял задание (а тут 3 глагола в повилительном наклонении), нужно обновить rate для id = 5 по ключу curr, а то, чего нет по ключу – добавить с id =5. Судя по структуре таблицы, нужно обратить внемание и на дату dt, для одного id и одной валюту дата не дублируется.

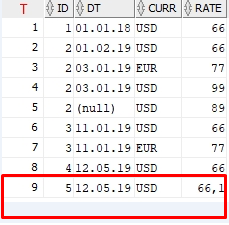
Решение:

SELECT \* FROM r

WHERE r.dt = '17.01.2019' OR r.dt = '12.05.2019'

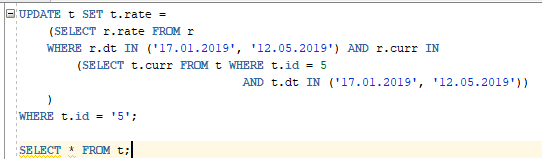


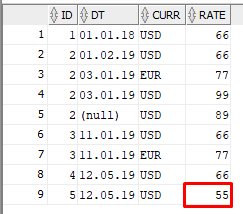
Вот эти записи из r мне необходимо обновить и вставить в t. Обновить данные для USD, а для EUR вставить новые.



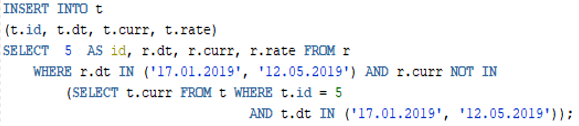
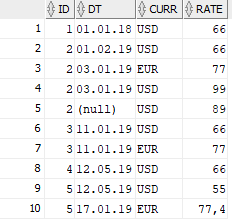
Решение 1:

Этим скриптом мы обновляем те котировки на валюты на указанные даты:





Теперь добавим те котировки на те валюты и даты, что не попали под предыдущие условия обновления. Осталось вставить эти записи с id = 5:

И так, полный скрипт:

UPDATE t SET t.rate =

(SELECT r.rate FROM r

WHERE r.dt IN ('17.01.2019', '12.05.2019') AND r.curr **IN**

(SELECT t.curr FROM t WHERE t.id = 5

AND t.dt IN ('17.01.2019', '12.05.2019'))

)

WHERE t.id = '5';

INSERT INTO t

(t.id, t.dt, t.curr, t.rate)

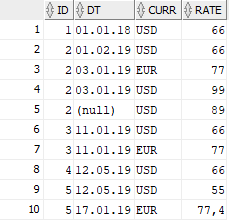
SELECT 5 AS id, r.dt, r.curr, r.rate FROM r

WHERE r.dt IN ('17.01.2019', '12.05.2019') AND r.curr **NOT IN**

(SELECT t.curr FROM t WHERE t.id = 5

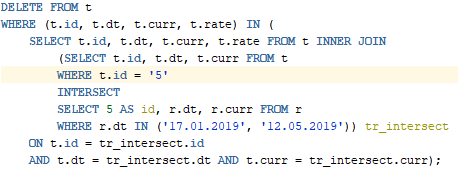
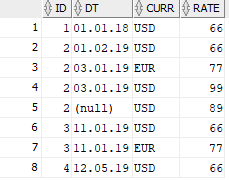
AND t.dt IN ('17.01.2019', '12.05.2019'));

Получаем в результате манипуляций обновлённую таблицу:

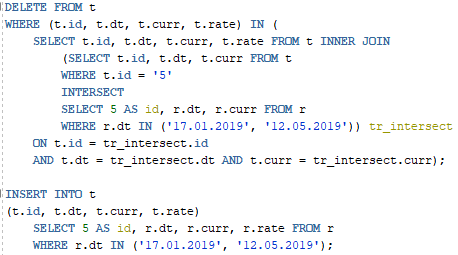


Решение 2 - А есть ли функция, что и вставит и перезапишет, например UPSERT? Ответ– нет. Окей, тогда просто пойдём другим путём сперва удалим то, что необходимо обновить и просто залём все данные, используем INTERSECT вместо множества условий:

Этой командой удаляем из t те записи, которые потребуется обновить, благодаря INTERSECT мы выбираем именно те записи, что подлежат перезаписи:

Теперь просто добавляем всё: что следовало перезаписать и что следовало добавить. Приведём сразу конечный скрипт:



На мой взгляд, используя манипуляции со множествами (INTERSECT, MINUS, UNION), меньше вероятность ошибки, чем с множеством AND, OR, IN. Код стал более прозрачен для чтения.

DELETE FROM t

WHERE (t.id, t.dt, t.curr, t.rate) IN (

SELECT t.id, t.dt, t.curr, t.rate FROM t INNER JOIN

(SELECT t.id, t.dt, t.curr FROM t

WHERE t.id = '5'

INTERSECT

SELECT 5 AS id, r.dt, r.curr FROM r

WHERE r.dt IN ('17.01.2019', '12.05.2019')) tr\_intersect

ON t.id = tr\_intersect.id

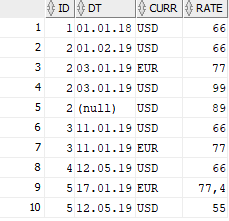
AND t.dt = tr\_intersect.dt AND t.curr = tr\_intersect.curr);

INSERT INTO t

(t.id, t.dt, t.curr, t.rate)

SELECT 5 AS id, r.dt, r.curr, r.rate FROM r

WHERE r.dt IN ('17.01.2019', '12.05.2019');



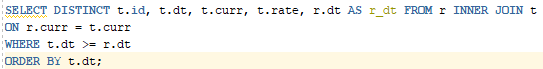
**Задача 5**

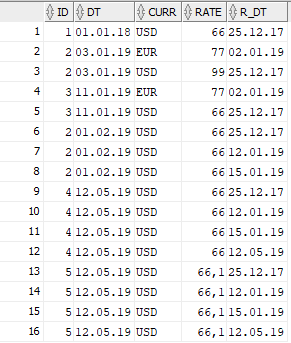
Напишите запрос, который обновит курсы валют в таблице t из таблицы r. Замечение! Курс берется на ближайшую предшествующую (<=) дату в разрезе валют. Т.е. если требуется обновить курс в таблице t за дату 01.01.2019, а ближайший курс в таблице r на дату 31.12.2018, то неоходимо брать его.

*Представьте не менее трех вариантов решения.*

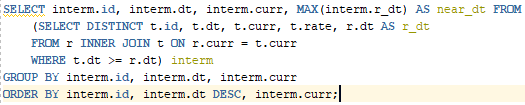
Решение:

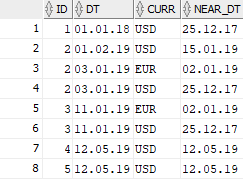
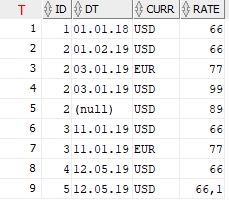
Шаг 1. Создаём промежуточную таблицу, где к каждой позиции таблицы t подобраны даты из r по ключу валюты и условию (<=) для даты :



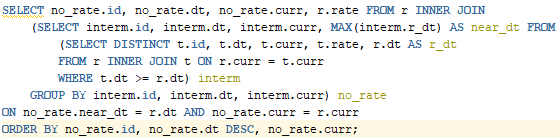


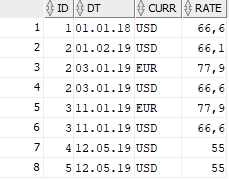
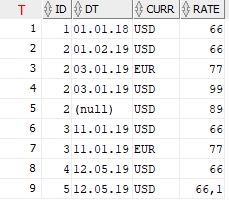
Шаг 2. Выбираем максимальные из них:



 🡺

Шаг 3. Таблица удовлетворяет условиям задания, осталось лишь из таблицы r взять соотвествующий валюте и дате rate. Вопрос, обновлять ли при этом дату? Поскольку в задании явно не указано – меняю только rate:



🡺

Осталось перезаписать таблицу, далее буду уже выкладывать различные варианты решения, вариируя способы перезаписи и порядок некоторых действий в формировании.

Решение 1 – базовое:

CREATE TABLE ark AS (

SELECT no\_rate.id, no\_rate.dt, no\_rate.curr, r.rate FROM r INNER JOIN

(SELECT interm.id, interm.dt, interm.curr, MAX(interm.r\_dt) AS near\_dt FROM

(SELECT DISTINCT t.id, t.dt, t.curr, t.rate, r.dt AS r\_dt

FROM r INNER JOIN t ON r.curr = t.curr

WHERE t.dt >= r.dt) interm

GROUP BY interm.id, interm.dt, interm.curr) no\_rate

ON no\_rate.near\_dt = r.dt AND no\_rate.curr = r.curr);

DROP TABLE t;

RENAME ark TO t;

Решение 2 – перезапись:

CREATE TABLE ark AS (

SELECT no\_rate.id, no\_rate.dt, no\_rate.curr, r.rate FROM r INNER JOIN

(SELECT interm.id, interm.dt, interm.curr, MAX(interm.r\_dt) AS near\_dt FROM

(SELECT DISTINCT t.id, t.dt, t.curr, t.rate, r.dt AS r\_dt

FROM r INNER JOIN t ON r.curr = t.curr

WHERE t.dt >= r.dt) interm

GROUP BY interm.id, interm.dt, interm.curr) no\_rate

ON no\_rate.near\_dt = r.dt AND no\_rate.curr = r.curr);

DELETE FROM t;

INSERT INTO t (t.id, t.dt, t.curr, t.rate) SELECT \* FROM ark;

Решение 3 – альтернативный синтаксический порядок и перезапись, убрал лишние слова:

CREATE TABLE ark AS (

SELECT no\_rate.id, no\_rate.dt, no\_rate.curr, r.rate FROM

(SELECT interm.id, interm.dt, interm.curr, MAX(interm.r\_dt) AS near\_dt FROM

(SELECT t.id, t.dt, t.curr, t.rate, r.dt AS r\_dt FROM t

INNER JOIN r ON r.curr = t.curr WHERE r.dt <= t.dt) interm

GROUP BY interm.id, interm.dt, interm.curr) no\_rate INNER JOIN r

ON no\_rate.curr = r.curr AND no\_rate.near\_dt = r.dt);

DELETE FROM t;

INSERT INTO t (t.id, t.dt, t.curr, t.rate) SELECT \* FROM ark;

Не смогу заставить работать upadate, возможно, надо было это делать циклом, но я их в SQL ещё ни разу не писал (хотя, похоже, надо начинать).

**Задача 6**

Напишите запрос, возвращающий строки из таблицы t (колонки id, dt, curr, rate), где курс максимальный. По одной любой строке с максимальным курсом на одну валюту (ключ - curr).

*Представьте не менее двух вариантов решения.*

Решение 1:

SELECT num\_t.id, num\_t.dt, num\_t.curr, num\_t.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.curr, MAX(t.rate) AS rate FROM t

GROUP BY t.curr) t\_max

ON t.curr = t\_max.curr AND t.rate = t\_max.rate) num\_t

WHERE (num\_t.num, num\_t.curr) IN (

SELECT MAX(num\_t.num) AS num, num\_t.curr FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

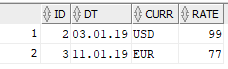
FROM t INNER JOIN

(SELECT t.curr, MAX(t.rate) AS rate FROM t

GROUP BY t.curr) t\_max

ON t.curr = t\_max.curr AND t.rate = t\_max.rate) num\_t

GROUP BY num\_t.curr)



Решение 2:

SELECT num\_t.id, num\_t.dt, num\_t.curr, num\_t.rate FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.curr, MAX(t.rate) AS rate FROM t

GROUP BY t.curr) t\_max

ON t.curr = t\_max.curr AND t.rate = t\_max.rate) num\_t

INNER JOIN

(SELECT MIN(num\_t.num) AS num FROM

(SELECT row\_number() over(ORDER BY t.id) num, t.id, t.dt, t.curr, t.rate

FROM t INNER JOIN

(SELECT t.curr, MAX(t.rate) AS rate FROM t

GROUP BY t.curr) t\_max

ON t.curr = t\_max.curr AND t.rate = t\_max.rate) num\_t

GROUP BY num\_t.curr) num\_min\_t

ON num\_min\_t.num = num\_t.num



**Задача 7**

От HR поступило задание: выдать список сотрудников, у которых зарплата больше, чем у непосредственного начальника.

*Структура данных на ваше усмотрение.*

Зададим структуру данных. Сотрудник может работать только в одном отдле. У каждого отдела может быть только один начальник. В рамках описанной ситуации глубокая нормализация не требуется. Обойдёмся двумя таблицами:

|  |  |  |  |
| --- | --- | --- | --- |
| employees | | | |
| id\_ emp | name\_emp | id\_dep | salary |

|  |  |  |
| --- | --- | --- |
| bosses | | |
| id\_dep | name\_dep | id\_ emp |

Напишем скрипт заполнения таблиц.

create table **employees**(

**id\_emp** number, -- Идентификатор сотрудника.

**name\_emp** varchar2(42), -- Имя сотрудника.

**id\_dep** number, -- Идентификатор отдела.

**salary** number -- Зарплата сотрудника

);

create table **bosses**(

**id\_dep** number, -- Идентификатор отдела.

**name\_dep** varchar2(42), -- Название отдела.

**id\_emp** number -- Идентификатор сотрудника - начальника.

);

commit;

Таблица всех сотрудников

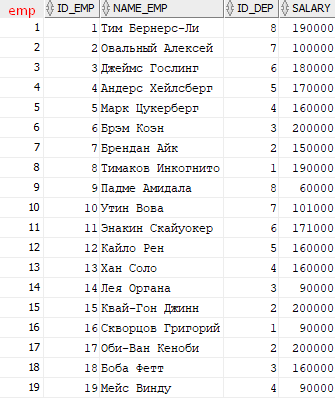
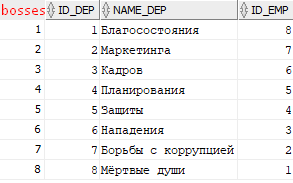
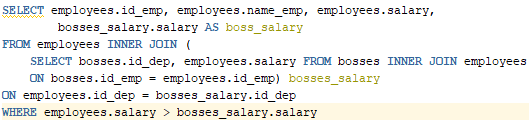


Таблица начальников отделов



Решение:

Запрос, выводящий необходимую информацию, id сотрудника, имя, зп, зп босса:



SELECT employees.id\_emp, employees.name\_emp, employees.salary,

bosses\_salary.salary AS boss\_salary

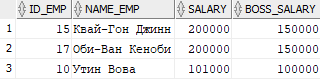
FROM employees INNER JOIN (

SELECT bosses.id\_dep, employees.salary FROM bosses INNER JOIN employees

ON bosses.id\_emp = employees.id\_emp) bosses\_salary

ON employees.id\_dep = bosses\_salary.id\_dep

WHERE employees.salary > bosses\_salary.salary



P.S.: Теперь кажется, что некоторые запросы я мог написать проще. Времени довольно ощутимо потратил на особенности сленга SQL в Oracle. Задания были интересные, хотя я чувствую, что большую часть из них я мог выполнить, используя циклы и функции. Но поскольку я был ограничен во времени, я предпочёл расписать запросы теми методами, которые я знаю хорошо. У меня есть ещё немного времени, возможно, всё же решу что-то через функции/циклы, прикреплю решения ниже.